Doc. No. 70751

AUTOMATED BUSINESS FORM INFORMATION ACQUISITION SYSTEM

CROSS-REFERENCE TO RELATED APPLICATIONS

[01] The present application is a continuation-in-part of co-pending U.S. Patent Application Serial No. 09/645,920 (hereinafter referred to as the '920 application), filed August 24, 2000, by S. Alhadad et al, entitled: "Automated Business Form Information Acquisition System," and further claims the benefit of co-pending U.S. Application Serial No. 60/252,033, filed November 20, 2000, by S. Alhadad et al, entitled: "Automated Business Form Information Acquisition System," each application being assigned to the assignee of the present application and the disclosures of which are incorporated herein.

FIELD OF THE INVENTION

[02] The present invention relates in general to communication systems and components therefor, and is particularly directed to an enhanced computer workstation-

associated call handling mechanism (which may be termed a 'Frequently Asked Question Software' (or FAQsoft)-based mechanism), for use by call handling agents employing a personalized response system (PRS)-based, and Forms-Based (target) software such as Customer Relationship Management (CRM) software in the course of servicing inbound or outbound telephone calls. The present invention is operative to automatically trigger actions, such as the automatic playback of one or more pre-recorded PRS phrases or other pre-defined actions at appropriate junctures during a call, in response to the call agent's interactions (via mouse or keyboard) with the target software. The phrases may include repetitive questions that are directly related to acquiring information from the client (the party calling or being called) for data entry into the software, or transmissions of required information to the client.

BACKGROUND OF THE INVENTION

[03] As described in the above-referenced '920 application, substantially all service facilities that are accessible by a telecommunication link, such as but not limited to a phone line, steer incoming calls to a computer workstation that is attended by an individual typically known as a call handler. Running on the workstation or an associated server is one or more application programs used by the call handler, call agent, operator and the like (hereafter referred to simply as a call handler), to process calls in

a manner that is efficient and satisfying to the calling party.

[04] As a non-limiting example, in the case of a catalog order center, a 'target' application program will typically graphically display an order or business 'form' on the call handler's terminal display. The displayed form contains respective information fields, through which the call handler sequentially 'tabs' and which are filled in by the call handler, based upon answers to a series of questions (such as name, address, phone number, item number, quantity, etc.), which the call handler speaks to the calling party. Because this repeated interactivity on the part of the call handler for multiple calls tends to be monotonous, and labor intensive, adversely affecting call handling efficiency, call handling facilities are continuously seeking schemes that will allow as much of the call handler's workload as possible to be automated.

[05] One scheme that has enjoyed widespread usage throughout the telecommunication industry for providing relief for a call handler or operator servicing incoming calls at an operator position is embodied in a device known as a 'personalized response system', and is the subject of U.S. Patent Numbers 4,697,282 and 4,623,761, assigned to the assignee of the present application and the disclosures of which are herein incorporated. In accordance with this personalized voice message retrieval and playback device, an operator will prerecord commonly used phrases in a

controllably accessible voice message storage and retrieval (VMSAR) mechanism (hereinafter referred to as a VMSAR mechanism. These phrases are then available for playback to a calling party (and simultaneously heard by the operator via the operator's headset) in the course of the operator's handling of a call. Playback may occur automatically, in response to initial signaling information contained in the incoming call, or manually by the operator manually invoking one or more inputs to the playback device. Because the prerecorded voice messages are played back in the voice of the operator, the calling party proceeds through the call in a straightforward and expeditious manner, perceiving that it is always the operator who is speaking.

[06] Although the patented VMSAR mechanism works very well for its originally intended usage, and in a limited scope of additional applications, it has been found that as the number of messages and degree of complexity of the operator's responsibilities increase, especially with the use of personal computer-based workstations running multiple windows-type programs, it becomes more difficult for the call handler to remember all the sets of keystrokes which access the various messages. While a list of keystroke combinations could be provided, having the operator look up which keystrokes are to be used would inherently delay the processing of the call. Even if the operator knows or can find what keystroke combination is to be used, there is the possibility of an incorrect or

unintended key operation, which may cause either no message or the wrong message to be played back. Realistically, therefore, the call handler cannot be expected to remember and use more than a relatively small number of keystrokes before confusion becomes an inhibiting factor.

[07] Pursuant to the invention disclosed in the '920 application, problems associated with conventional call handler workstations are effectively obviated by a call handler-interactive mechanism, that automatically triggers playback of selected voice messages that have been pre-recorded via a personalized response system-based voice message storage VMSAR mechanism, in response to a call handler performing a prescribed interaction, such as a tab keystroke or mouse click, with one or more elements, such as 'control objects' displayed on a GUI of the call handler's terminal. To provide voice communication connectivity with an external network, the workstation may be coupled to a private branch exchange (PBX) or switch. The PBX or switch is also coupled to the VMSAR mechanism through the voice terminal.

[08] The GUI generated on the workstation display may comprise, for example, a 'windows' type user interface, generated by an underlying call handling application program that may be running on the workstation's computer or on a separate server. The GUI typically contains a plurality of fields, buttons, and the like, in which information may be keyboard entered by the call handler, in

the course of servicing a call, such as one placed to a catalog order center. The '920 invention reduces the workload on the part of the call handler, by integrating with or into the target program software a GUI-associated, event-driven VMSAR mechanism.

[09] The GUI is continuously monitored for the occurrence of prescribed events - corresponding to the call handler performing a prescribed interaction with a selected control object in the target program. As non-limiting examples, such GUI interaction may include placement of the display screen cursor arrow anywhere within the perimeter of the control object and a click of the 'left' mouse button, depression of the 'enter' key on the keyboard when a displayed reticle is placed upon a control object, operation of the 'tab' key that moves the cursor from object to object in the GUI, and the like.

[10] When a desired event is raised, a digital (voice message playback) control message packet is generated, to specify a message number of a selected voice message to be played back by the VMSAR mechanism to the calling party. Each personalized voice message is predefined in accordance with the respective control objects of the displayed business form to be completed by the call handler. In large majority of instances, the phrases are brief questions that prompt the calling party to provide information to be entered into a particular field of the GUI.

[11] In order to expedite the processing of a call, a respective control message packet is transmitted to the VMSAR mechanism immediately in response to a desired interaction of the call handler with a GUI control object. In a typical case, the control object of interest would be an empty information field, so that the VMSAR mechanism would be immediately triggered to generate the voice message associated with that field. However, the field of interest may already contain alpha-numeric text, such as may be obtained from calling ID capability of the telecommunication service provider, that has allowed the call handler application program to have already preliminarily entered text information in one or more fields prior to the call handler beginning a dialogue with the calling party. In this case, the call handler may simply confirm the accuracy of the filled-in information with the party. It will then be up to the call handler to either speak to the party, or interactively advance the cursor, for example, to the next empty field, which raises an event that triggers an associated voice message playback from the VMSAR mechanism.

[12] In addition to triggering the playback of a prescribed voice message by the VMSAR mechanism, the '920 invention may also cause the words of that voice message to be displayed in a preselected dialogue/text box within the GUI, to confirm to the call handler that the intended message has been played back to the caller. A visual

indication of the state of a control object may also be supplied, by means of a label, such a solid color region, placed immediately adjacent to the control object, to indicate the presence of an associated voice message. When the VMSAR mechanism is triggered, the background (e.g., color) of the label is changed, to indicate that the message has been played back. The label may then act as a button to be selectively triggered by the call handler.

SUMMARY OF THE INVENTION

[13] As described briefly above, the present invention is directed to an enhanced computer workstation-associated call handling mechanism, that may be readily employed by a call handling agent to automatically trigger actions, such as but not limited to the automatic playback of one or more pre-recorded PRS phrases or other pre-defined actions at appropriate junctures during a call, in response to the call agent's interactions (via mouse or keyboard) with a personalized response system (PRS)-based, and Forms-Based (target) software such as Customer Relationship Management (CRM) software in the course of servicing inbound or outbound telephone calls.

[14] Screen markers on the Graphical User Interface (GUI) elements of the Target Software make the Call Agent aware of the GUI elements (e.g., field boxes, buttons, pictures, etc. otherwise known by one schooled in the art as □controls□) are associated with a pre-defined action (or

series of actions) and have been enhanced by the invention. Variations in the screen markers represent different types of automated actions that manipulation of the GUI elements will execute.

[15] The present invention also provides the ability to manually execute pre-defined actions (playback of PRS Phrases and other actions), as by operating specific keys or key combinations (Hotkeys) or mouse-clicking configurable buttons (FAQsoft Buttons) housed within a Windows Appbar application (FAQsoft Deskbar). The invention enables a Call Agent, Supervisor or other administrator to develop Responses (pre-defined single or multiple actions), associate the Responses to a Hotkey or FAQsoft Button, and trigger Responses using hotkeys or FAQsoft Buttons during the normal progress of a telephone call.

[16] A fundamental aspect of the present invention is its ability to respond to user interactions with predefined GUI elements (Controls) in an existing Target Application, without having to modify the source code of the Target Software. Instead, hooking and sub-classing techniques made available in WinOS are employed to change the behavior and look of these Target Applications to provide a higher level of automation for Call Agents, without requiring any changes to the Target Application itself.

[17] The present invention enhances existing software by providing automated playback of pre-recorded phrases; however, this action can be expanded to include any action

the computer is capable of executing via the use of scripts. Screen Indicators appear on Controls that have been configured for Responses.

BRIEF DESCRIPTION OF THE DRAWINGS

[18] Figure 1 is a block diagram of a computer-based call handler employing the FAQsoft mechanism of the invention;

[19] Figure 2 shows a window programming tool;

[20] Figures 3 and 4 show SUBCLASSING programming tools;

[21] Figure 5 shows a HOOKING programming tool;

[22] Figure 6 is a flow chart showing the process of invoking and running the FAQsoft mechanism of the present invention;

[23] Figure 7 depicts a FAQsoft□s Tray Application icon;

[24] Figure 8 shows a Tray Menu;

[25] Figure 9 shows a Login window;

[26] Figure 10 shows an Appbar Application window;

[27] Figure 11 shows a Phrase Manager window;

[28] Figure 12 shows a Hotkey Manager Window;

[29] Figure 13 is a dialog box displayed by the Hotkey Manager window;

[30] Figure 14 shows a Configure Manager window;

[31] Figure 15 shows an Add Actions Dialog - Readback Contents window;

[32] Figure 16 shows an Add Actions Dialog - Silence window;

[33] Figure 17 shows an Add Actions Dialog - Execute script

window;

[34] Figure 18 shows a Phrase/Response List in a Quick Configure Window;

[35] Figure 19 is an Options Dialog window;

[36] Figure 20 shows a User Manager window with a list of users;

[37] Figure 21 shows a User Manager window with User Info;

[38] Figure 22 shows a User Manager window with Access Rights;

[39] Figure 23 is a flow chart showing the steps of the Monitor Module process flow;

[40] Figure 24 shows a normal Outlook Contact window before running the FAQsoft routine;

[41] Figure 25 shows an Outlook Contact window after being configured with FAQsoft and with FAQsoft running;

[42] Figure 26 shows the process sequence when FAQsoft is in Configure Mode;

[43] Figure 27 shows a Configure Info Dialog window;

[44] Figure 28 shows a Configure Edit window;

[45] Figure 29 shows an original context menu on a monitored text box; and

[46] Figure 30 shows a substituted context menu on a monitored text box.

DETAILED DESCRIPTION

[47] Before describing in detail the automated business form information acquisition system of the present invention, it should be observed that the invention resides primarily in the augmentation of the software employed by the call handler's computer workstation, being operative to automatically trigger actions, such as the automatic playback of one or more pre-recorded PRS phrases or other pre-defined actions during a call, in response to the call agent's interactions (such as through a mouse click or keyboard action) with the target software. The phrases may include repetitive questions that are directly related to acquiring information from the party calling or being called for data entry into the software, or transmissions of required information to the client.

[48] Consequently, the invention has been illustrated in the drawings in readily understandable block diagram, flow chart and associated GUI display format, which show only those specific details that are pertinent to the present invention, so as not to obscure the disclosure with details which will be readily apparent to those skilled in the art having the benefit of the description herein. Thus, the illustrations are primarily intended to illustrate the major components of the invention in a convenient functional grouping, whereby the present invention may be more readily understood.

[49] Attention is initially directed to Figure 1, which shows the main components of the FAQsoft mechanism of the invention. It is comprised of a Personal Computer (PC) 10 running Microsoft Windows Operating System OS (e.g., Win95 and later), but can also be implemented in similar event-based Graphical User Interface Operating Systems. The PC 10 is coupled to conventional input/output devices, such as a display device 22, a pointing device 23, a keyboard 24 and other ports 25. The system can include a server 26 to archive files of personalized voice recordings (Audio Files) and to store the FAQsoft database. It is connected to a PRS 21 via a port for data communications, or may be connected to the sound card I/O ports or other ports for voice communication with a telephone system 29.

[50] The PC 10 is used to run Target Applications typically used in Call Centers for taking orders or collecting information from a client. A user interacts with the Target Application's Graphical User Interface (GUI) 30 using a keyboard and/or mouse or other input device. The GUIs may include several different types of windows. At the top level are windows that have title bars and are called Top Level Windows. Other GUI elements hosted by Top Level Windows are commonly referred to as Controls.

[51] Controls may include a variety of different windows (e.g., Forms, Labels, Text boxes, Buttons, Check boxes and so on). All windows - Top Level windows and Controls - have procedures that define their behavior commonly referred to

as Windows Procedures. Using the mouse 23 or keyboard 24, the User may interact with Controls and thereby cause the operating system 15 to generate messages (commands) directed at these Controls. The Target Application 14 responds according to its programmed Windows Procedures.

[52] The FAQsoft mechanism 11 monitors messages sent to these Controls and modifies their behavior by substituting or appending another procedure to the Control's original Windows Procedure. Only Controls in applications that have been explicitly configured for FAQsoft monitoring will exhibit these added functionalities.

[53] Data for FAQsoft resides in a FAQsoft Database 27, either in the local hard drive 20 or at a server computer 26. The database stores information mainly on Control/Response association, Phrases, Hotkeys and Users. Information stored for each User includes controls to monitor in the target application, which audio files of recorded phrases to use (only audio files created in that User□s voice) Audio File directories and that User□s Hotkeys List, Deskbar settings and User access rights.

[54] Before describing the details of the methodology of the present invention, a number of programming tools will be reviewed in order to facilitate the detailed description of the invention to follow.

WINDOW (Figure 2)

[55] In an Event based GUIs Environment like Microsoft Windows, application screens are built using various

'windows', such as Edit box, Labels, Frames, Checkboxes, ComboBoxes and etc. These objects are actually windows with different styles and can be contained within other windows. Such windows are also known as 'Controls'. Windows that have a title bar are usually known as Top Level windows. Each window has a function to tell it how to behave when a user interacts with it. When a mouse pointer is clicked on a control, the OS sends a message to that control's windows function. Almost all interactions by the user with application software generates one or more messages that are queued within the OS message queue and are forwarded to the respective windows functions. The OS also provide tools (APIs) to allow programmers to trap these messages before they reach the intended windows function.

SUBCLASSING (Figures 3 and 4)

[56] Subclassing is based on the idea of intercepting messages by changing the function that is associated with a window, forcing messages going to a window to first execute the substituted code instead of the function originally assigned to a window. This provides the option of calling the original window function, if desired.

[57] Subclassing can only handle one window at a time, so that it is necessary to individually subclass each window to be monitored. This is actually desirable for applications, such as FAQsoft, so that only messages of concern are received and only from controls of concern. This translates to less impact on overall system

performance.

HOOKING (Figure 5)

[58] In order to view or interact with system wide messages, it is necessary to employ another technique call 'Hooking', which allows for system-wide or application-specific monitoring of events. The SetWindowsHookEx api contains special calls to trap and intercept mouse, keyboard and other system messages across all system process or specific process threads. Figure 5 shows the WinOS message flow and some possible hook placements. A mouse hook WH_MOUSE is selectively placed to intercept a mouse message to a system message queue. A keyboard hook WH_KEYBOARD is selectively place to intercept a keyboard message to the system message queue. Figure 5 also shows a WH_GETMESSAGE hook placed to intercept a message transmitted from a threads message queue to the window function.

[59] The principal requirements of the FAQsoft mechanism of the invention are as follows:

- Uniquely identify all GUI objects on a Forms based application (Target Application)

- Draw and persist screen indicators on GUI elements controls on the target application to indicate that it is being monitored. (FAQsoft Enabled)

- Associate one or more actions to take place when specific interactions is initiated by the User with these monitored GUI elements.

- Respond to user manipulations or interactions with the selected GUI object by executing the actions defined previously.

- Save the configuration for each target application so it may be recalled and used to accordingly.

- Support monitoring a multiplicity of Target Programs concurrently.

- Must not allow a perceivable degradation of system performance.

- Provide Tools to associate Responses with GUI objects in Target Applications.

[60] Attention is now directed to Figure 6, which is a flow chart showing the process of invoking and running the FAQsoft mechanism of the present invention. FAQsoft is designed to run silently as a background process and will auto start when the computer is booted up. When FAQsoft is first started at step 100, it installs an icon into the system tray at step 110. System tray icons are icons that reside at the bottom right area of the desktop. They are usually reserved to represent applications that run in the background. Figure 7 depicts the FAQsoft□s Tray Application icon □White Quotes on Red□ in the system tray after it is installed by this step.

[61] Next, in step 120, the Tray Menu is installed (FAQsofts menu system shown in Figure 8) to provide access to other modules and areas in the FAQsoft routine. This hierarchical menu is activated when the user right clicks

on the tray icon. To access the menu, the user must first be logged in.

[62] The Login Dialog is shown in Figure 9. In step 130, the FAQsoft routine first gets the User List from the database and the name of the last user from Windows Registry 28. The Windows Registry is a specialized memory area reserved by WinOS for persisting values for application settings. A Login Dialog with a dropdown list of valid users with the last User Name showing in the Login field of the Dialog is then presented in step 310.

[63] To operate FAQsoft, there must always be a valid user logged in (step 140), since it needs to play back personalized voice recordings to the phone system, and therefore must know and prepare the current user's audio files. Upon successful login, the current user name is saved in Windows Registry in step 150, and the user's specific information (User Profile) is retrieved and loaded in step 160.

[64] The User Profile consists of all information required to customize the FAQsoft program for that user. It includes values from the Windows Registry, such as settings for the Deskbar, location of the database and user's pre-recorded voice files and data from the FAQsoft database. This includes Configure Records, Phrase Records and Hotkeys Records necessary for execution of the FAQsoft routine. The Configure Records include information that associates Controls with one or more actions to execute (a Response).

The Phrase Records define phrases to be recorded or otherwise used by each Call Agent, and contain phrase name, phrase description and file location for each phrase. The Hotkey records relate each Hotkey combination with a Response.

[65] In step 170 the FAQsoft Deskbar application is started and displayed as per settings from the Registry shown in Figure 10. It contains Soft Buttons that are associated with pre-built Responses and a text area that displays the Response text from the Configure Record.

[66] If the audio files are not available locally, the FAQsoft program will retrieve them from the Server or have the user record them first before continuing in step 180. The Phrase Manager shown in Figure 11 will be presented if listed phrases have not been recorded. The Phrase Manager provides all necessary tools including a Sound Editor to facilitate recording and editing of phrases.

[67] The Monitor Module 13 is then loaded in step 190. As FAQsoft's monitoring engine this makes it possible to change the behavior and appearance of selected Controls within the Target Application. The Monitor Module 13 is implemented as an ActiveX Server that runs as a faceless application in the background. It communicates with the FAQsoft's Tray Application via event messages. This module will be discussed in detailed below.

[68] The Monitor Module 13 raises events in step 200 that are consumed by the FAQsoft Tray Application. These events

become commands containing Responses for the FAQsoft Tray Application to execute. The FAQsoft Tray Application parses each Response in step 210, to determine the discrete actions required, and then executes each action sequentially until all actions contained in the Response have been executed. Valid actions supported by the current embodiment of FAQsoft include:

- If action = Play (step 220), e.g., Play:Greeting, then proceed to play that Phrase (step 230).

- Play the phrase using the selected PRS mechanism and continue to parse action list.

- If action = Silence (step 240), e.g., Silence: 1000) then pause for specified time (number of milliseconds) (step 250) and continue to parse action list.

- If action = Readback (step 26), e.g., Readback:D445-4564B) then proceed to process the text to Readback (step 270), if recordings exist for the alpha numeric phrases.

- Play the phrase using the selected PRS mechanism and continue to parse action list.

[69] If action = Execute (step 280), e.g., Execute: ToggleDeskbar, then Execute predefined script or function specified (step 290) and continue to parse action list.

[70] The user may access other modules in FAQsoft using the Tray Menu of the FAQsoft Tray Application. Selecting a menu item will start its corresponding module or component.

[71] Steps 300 - 480 of the routine list the more significant menu items as follows:

- Login step 300 loads the Login Module (step 310) that displays the Login Dialog to change user.

- Phrase Manager step 320 loads the Phrase Manager Module (step 330) that displays the Phrase Manager Window, shown in Figure 11. The Phrase Manager has three main functions:

1- Maintenance - including adding, deleting and editing phrase records;

2- Recording audio recording of phrases; and

3- Quality Review of recorded phrase by supervisors.

- The Hotkeys Manager step 340 loads the Hotkeys Manager Window shown in Figure 12. The user can drag and drop Response Items from the Configure Manager into this window. The Hotkey Manager will display a dialog as shown in Figure 13 to allow the user to designate a unique key combination that will be used to trigger the playback of a phrase or execution or a Response. The Allow Default Action checkbox will permit pre-existing hotkeys in other applications to be executed concurrently. Double-clicking a Hotkey item shows the Hotkey dialog for further edits. When the save button is click, FAQsoft persists the new Hotkey item and registers the Hotkey.

[72] The Configure Manager step 360 loads the Configure module (step 370). When started, this module displays the Configure Manager window shown in Figure 14, which serves the following functions:

1- Changes FAQsoft to operate in Configure Mode. This mode of operation allows the selection of controls in the Target Application by using a drag and drop action from either a Phrase item or a Response item for the purpose of associating that control with the dragged item;

2- Create new Response Records. A Response Record may contain:

- Concatenated phrases (a series of phrases) by dragging and dropping phrases items from the Phrase List to the Action List area;

- A list of other actions by clicking on the Add button and selecting one of the available actions from the Add Actions Dialog. (See Figures 15-17.)

[73] Add Actions Dialog - Readback Contents Figure 15: Reads back alpha numerically the text contents of the currently selected control or from another control.

[74] Add Actions Dialog - Silence Figure 16: Pause for a duration equal to the value entered in millisecs.

[75] Add Actions Dialog - Execute script Figure 17: Select a script to execute from a list of pre-built scripts.

[76] The Quick Configure step 380 displays the Quick Configure Window (step 390), shown in Figure 18. This window lists the combination of Phrase items and Response items and takes up relatively little display screen area. It also causes the FAQsoft routine to operate in Configure Mode, so that the items can be dragged and dropped onto controls in Target Applications.

[77] The Options step 400 calls the Options Dialog shown in Figure 19 that has different tabs to define settings for the following:

1- General: Choice to use the FAQsoft Monitoring Engine or Microsoft Active Accessibility for monitoring.

2- PRS: which PRS model to use.

3- Phrase Manager: Settings for default user voice file locations.

4- Database: Default location and other information related to FAQsoft's database.

5- Keyboard: Keyboard Layout Settings.

[78] The Users step 420 calls the Users module (step 430), which loads the user table from the FAQsoft Database and displays the User Manager Window shown in Figures 20-22. The user database stores registered users of the FAQsoft system and maintains the applications and phrases that each user is setup to use with the FAQsoft routine. It also stores the Security Rights for access to different parts of the FAQsoft routine.

[79] The Exit step 440 saves all settings to Windows Registry (step 450), unloads instances of all modules including the Monitor component (step 460) and removes the system tray icon (step 470). The FAQsoft routine completes all necessary cleanup before quitting the tray application (step 480).

[80] Figure 23 shows the Monitor Module process flow.

[81] When a user logs into FAQsoft, in step 500 the Monitor

component updates that users lookup tables 13 in memory consisting of Configure Records and Hotkey Records from the database 27. A specific user's Configure Records and Hotkey Records are known as the Configure List and Hotkey List.

[82] Each Configure Record consist of the following fields:

[83] Hwnd: The control's window handle that is dynamically updated when a control is initially found in an active Top Level Window.

[84] Control Name: a name for a configured Control created by FAQsoft when possible by extracting text from the control or the Control's closest label.

[85] Static Information: a collection of inherent attributes about a control that uniquely identifies it. (A detailed description is presented below.)

[86] Event: the user action to monitor.

[87] Condition: state of the Control or a system value to check, e.g. is the Control value = Empty?

[88] Response: one or more actions to execute.

[89] Response Description: A textual description of the response. This usually contains the actual script that the User is to record. This is displayed in the Deskbar when the response is executed.

[90] The Hotkeys List also associates each defined hotkey combination in the list with a corresponding Response to execute. It contains the following fields:

[91] Keycode: the numerical value of the key combination.

[92] KeyCombo: the textual representation of the hotkey

(e.g., Ctrl+F1).

[93] Response: one or more actions to execute.

[94] Response Description: A textual description of the response. It usually contains the actual script that the User is to record. This is displayed in the Deskbar when the response is executed.

[95] In order to enable the FAQsoft routine to monitor other applications, hooks are installed (step 510) which trap Windows messages sent to these applications. These hooks are implemented as a Windows dll (Dynamic Link Library) file 12; two separate system wide hooks are installed using the SetWindowsHookEx function call. The Message Hook installed monitors only the WM_ACTIVATE message, in order to monitor focus changes occurring in Top Level Windows only. The Keyboard Hook is installed to implement the Hotkey function. Hotkey combinations found in the Hotkeys List are then registered in step 520. The Keyboard Hook installed traps all registered hotkey messages.

[96] By looking at the WM_ACTIVATE message, the FAQsoft mechanism is informed whenever the user changes the active Top Level Window. When a WM_ACTIVATE message is received (step 560), FAQsoft enumerates the Controls contained within the active Top Level Window (step 570) using Win32 API call EnumChildWindows. It tries to match items in the Configure List with Controls that exist within the currently active Top Level Window (step 580). If a match is

found, that Control is subclassed (step 590), the hwnd field of the matching items in the Configure List (Configure Items) is updated with the control's window handle (hwnd) (step 600). User interaction with that control is monitored once it has been subclassed.

[97] With respect to static information, the FAQsoft program extracts embedded information from a control and concatenates them to form a key that uniquely identifies each selected control (Control Static Information). This key is made up of information that remains static between multiple instances of the same Control and is generated in a similar manner as the Spy++ program found in Microsoft Visual Studio toolset. It includes the Control's Application Name, Class Name, Class ID, Parent Name, Parent ID, Location and a few other specific characteristics of the control obtainable via Win32 API calls that require the window handle (hwnd) value of a control. A Control's Static Information does not include the hwnd of the Control because this value will be different every time the Target Application creates that control.

[98] During a previous Configuration process the same key (Static Information) would have been generated when a user selects the Control. This is stored in the Static Information field of the Configure Record. FAQsoft also attempts to supply a descriptive name for that Control and stores it in the Control Name field of the Configure Record. This name is generated differently depending on the

selected Control's Class Type. For Controls with a Static Text property, such as Labels, Buttons, Dialog boxes and Icons, their embedded text is used and the control type is appended after the name. For Controls with editable text, the FAQsoft routine finds the closest label situated to the left or top of the selected control, extracts that Label's text value and appends an abbreviated control type description to the Label text to form the Control Name.

[99] For example, if the Save button is selected, the FAQsoft routine will use the button name and postfix the control type to the name and the value "Save_btn" will be stored as the Control Name in the Configure Record. As another example, if the Job Title field on a form is selected, the FAQsoft routine will recognize this as an editable text control and proceed to locate the Label for that field and store "Job Title_txt" as the Control Name value in its Configure Record. The Configure Record's Control Name field is used as a key principally in search operations called by the User.

[100] When a match is obtained, the FAQsoft routine updates the Configure List 13 with the current window handles after subclassing these controls so that subsequent searches will use the hwnd value instead of Static Information value. This is desirable since FAQsoft has to build the Static Information for each control before it can perform a search operation, making Static Information based searches a more expensive process. Search operations using the hwnd value

are possible, since window handles are guaranteed to be unique by WinOS for the life of the Control.

[101] As pointed out above, subclassing is a well known programming tool that is used to monitor and modify the behavior of Controls and is similar to hooking. However, subclassing is less intrusive and is targeted only at individual Windows or Controls. The hwnd (window handle) of each Control must be known before it can be subclassed.

[102] The FAQsoft routine's subclassed Controls only monitor messages supported in the Events field of the Configure List. It traps the messages listed below to detect the following user actions. It should be observed that other messages may be added to expand the system's monitoring capabilities.

[103] Windows Messages

WM_DESTROY: user has closed the Top Level Window.

WM_SETFOCUS: user has either tabbed into or clicked on the control.

WM_KILLFOCUS: user has tabbed out of or clicked on another control.

WM_PAINT: control requires redrawing of screen indicators.

WM_ERASEBKGND: control requires redrawing of screen indicators.

Mouse Messages

WM_LBUTTONDOWN: Left button single click.

WM_LBUTTONDBLCLK: Left button double click.

WM_RBUTTONDOWN: Right button single click.

[104] Once a Control is subclassed, the FAQsoft routine will be aware of user interactions with that control. For example a tab into an edit box will cause the edit box to receive focus. WinOS sends a WM_SETFOCUS message to the edit box which is intercepted (via subclass mechanism) and recognized by FAQsoft as one of the monitored message (step 620). The FAQsoft routine then compares the editbox's hwnd with hwnds in the Configure List (step 630). If a match is found, the Event parameter is compared to determine a valid match and raises an event (command) for the Tray Application to consume (step 640) as illustrated by the following pseudo-code example:

```
'If hwnd matches then user is manipulating a monitored control
    for each currentHwnd = ConfigItemHwnd
    if currentEvent=ConfigItemEvent then
    currentControlText=read text contents of current control if any
        RaiseEvent (currentControlText,ConfigItemResponse)
        Exit For
        end if
        next.
```

[105] The Tray Application (step 650) executes the Response if the Condition parameter is met (step 201). It operates as a separate thread so as not to impede the Monitoring Module's performance. Other Mouse messages received from

monitored Controls are treated similarly (step 621).

[106] When a WM_DESTROYmessage is received (step 622), it means that the Control is being closed by the Target Application and the subclass for that control must be removed. FAQsoft removes the subclass as the destroy message is received from each Control (step 660) and resets the hwnd value to zero for related Configure items in the Configure List.

[107] The WM_PAINT or WM_ERASE message is received (step 623), whenever repainting is require for that control. The FAQsoft routine will draw screen markers on the control in step 670 to indicate to the user actions that will cause the FAQsoft routine to respond (additional behavior for that Control). As a non-limiting example red geometrical shapes painted on different locations on the control may be used to indicate the different events to which the Control has been configured to respond. The following table lists the shape and events relationship:

Screen Indicator	Events or user action
Top Left triangle	On receive focus
Top Right Triangle	On leave focus
One pixel border	On Single click
Two pixel border	On Double-Click

[108] Figure 24 shows a normal Outlook Contact window before running the FAQsoft routine and Figure 25 shows the same

window after being configured with FAQsoft and with FAQsoft running. The screen indicators provide a visual cue for the Call Agent to anticipate certain behavior when he or she interacts with FAQsoft's enhanced Controls in the Target application.

[109] It is possible to painting controls in the Target Application's window since the Device Context (dc) can be obtained for each control in question, adding graphics to it using conventional drawing techniques. With proper graphic resource management, the user will not perceive any delay in the presentation of these screen indicators.

Hotkeys

[110] Hotkeys are registered, so that the keyboard hook will receive an indication (message) in step 530, when a registered key combination is depressed. Upon receipt of a hotkey press, the key code value is compared to Keycodes in the Hotkeys List in memory (step 540). When a match occurs, an event is raised (step 640) and the associated Response is sent to the Tray Application for execution.

[111] When the Monitor Module receives an Exit message from the Tray Application (step 680), it goes through a clean up process (step 681). It un-register all hotkeys, unsubclass any controls that may be still be subclassed and removes any hooks before exiting (step 682).

Control Configuration

[112] The FAQsoft routine operates in Configure Mode (Design Mode) whenever the Configure Manager (Figure 14 or Quick

Configure Figure 18) window is displayed. The Configure Mode is a state in FAQsoft where selecting and associating Response records to Controls is conducted. Figure 26 shows the process sequence when FAQsoft is in Configure Mode.

[113] When Configure Mode is enabled in step 810, the user is able to click and drag Phrase and Response items from their respective lists and drop them onto Controls in Target Application windows. Phrase items dropped on controls automatically generates a Response Record with the value Play: <PhraseName>. When the user clicks on either a Phrase or Response item, a MouseDown event (step 820) on the List (grid Control) that host these items (Figures Figure 14 and Figure 18) triggers a SetCapture API call (step 830), which causes the mouse pointer to be tracked across the screen (step 831). A timer is used to periodically call GetWindowFromPoint( ) to get the window handle (hwnd) of the Control below the mouse pointer (step 832) and highlight the Control under the mouse.

[114] The control's bounding rectangle is obtained with a call to GetWindowRect() once the hwnd is found and a yellow border drawn (Figure 18) on this rectangle (step 833) to indicate to the user the currently selected control (highlighted control). When the mouse pointer moves over to another Control, the framed yellow rectangle is erased from the old Control and drawn over the new control. The process continues until the user releases the mouse button.

[115] When the user releases the mouse button, it generates a MouseUp event (step 840). The FAQsoft routine then erases the highlighted rectangle in step 850, and calls ReleaseCapture( ) in step 851, to stop mouse pointer tracking. The control's Static Information is obtained in step 852, and matched against the same Configure List in Monitor module (step 853). If the selected control already exist (screen indicator present) in the Configure List, then all Configure Items associated with that control are recalled (step 870). These items are then displayed in step 880, together with a newly created Configure Item (step 871) to reflect the latest addition in the Configure Info Dialog window (Figure 27).

[116] The FAQsoft routine examines the Event and Condition values of the existing Configure items to try and select the next most logical Event/Condition combination for the newly created Configure Item. Each control can only accept unique Event/Condition combined values. For example, if a control is already configured with the following record:

ControlName = CreditCard #_txt

Event = OnEnter

Condition = Empty

Response = AskCardNumber

[117] And the user drags in a Response Item called "VerifyCardNumber" into the same Control, the FAQsoft routine will automatically fill the Condition and Event

fields with the following values for the new Configure Item:

ControlName = CreditCard #_txt --> CreditCard #_txt

Event = OnEnter --> OnLeave

Condition = Empty --> Not Empty

Response = AskCardNumber --> VerifyCardNumber

[118] The FAQsoft routine assumes the user will want to have the second Response item triggered when the user leaves the CreditCard field (OnLeave), and only when there is some value typed into that field (Empty). The user may change these values by either double-clicking on the item or selecting an item and clicking on the Edit button (step 881), which opens the Configure Edit window (Figure 28).

[119] The FAQsoft routine allows the Event and Condition values to be changed, as long each item's Event and Condition combined value remains unique for each Control.

[120] When the user exits the Configure Dialog with the Save button, the FAQsoft routine updates the current Configure List (step 883) and maintains all changes made to the Database 27 (step 884). If the user exits by clicking on the Cancel button, then no change to the Configure List , Database or control is made.

[121] The FAQsoft routine uses the Monitor module to help make it aware of Controls already configured within Target Applications. When the user right-clicks on a monitored control while FAQsoft is in Configure Mode, the Controls context menu is substituted with one supplied by the

FAQsoft routine. Figure 29 shows the original context menu and Figure 30 shows the substituted one. When the Monitor receives a WM_RBUTTONUP message while in Configure Mode, it raises the ShowContextMenu event (step 891), to inform the Tray Application and does not let that message go to the intended Control.

[122] Upon receiving the event, the FAQsoft routine replaces it with its own Context Menu and displays it (step 892) at the position where the click occurred. The replaced context menu contains only two items, Clear All and Edit. When the user selects Clear All menu item (step 893), all Configure items related to the clicked control will be removed from the Configure List (step 895). The Database is updated and the Control's subclassed (step 896) is removed. When the user selects Edit menu item, the FAQsoft routine loads the Controls Configure items and displays them in the Configure Info Dialog shown in Figure 27.

[123] As will be appreciated from the foregoing description, by using hooking and sub-classing techniques to respond to user interactions with predefined graphic user interface elements of an existing target application, the FAQsoft mechanism of the present invention avoids having to modify the source code of the target software. This enables the invention to provide a higher level of automation for call agents, without requiring any changes to the target application itself. As such, the invention may be readily employed by a call handling agent to automatically trigger

various call handler actions, including automatic playback of pre-recorded personal response type phrases or other pre-defined actions at various instances during a call, in response to the call agent's interactions (such as via a mouse or keyboard) with a personalized response system-based, and forms-based software in the course of servicing telephone calls.

[124] While I have shown and described an embodiment in accordance with the present invention, it is to be understood that the same is not limited thereto but is susceptible to numerous changes and modifications as known to a person skilled in the art, and I therefore do not wish to be limited to the details shown and described herein, but intend to cover all such changes and modifications as are obvious to one of ordinary skill in the art.